

ECHO REST Search Guide

Overview

The purpose of this document is to guide new and existing ECHO client developers through the process of searching ECHO's metadata catalog their data via our REST interface. It is intended to be a hands-on, step-by-step introduction to the process. Any comments or questions regarding the contents of this document should be directed to echo@echo.nasa.gov

Assumptions and Pre-requisites

This document is targeted at a technical audience and is intended to be a high-level roadmap for search and ordering implementations. While much of this material should be accessible to novice ECHO users, there are a few technologies and configurations that will prove useful to have in your tool belt to follow along with this document.

1. A working knowledge of REST concepts (http://en.wikipedia.org/wiki/Representational_state_transfer)
2. Familiarity with ECHO and the EOSDIS User Registration System (URS) (necessary for ordering/finding certain data)
3. Familiarity with the ECHO10 format at both the dataset and granule level.
4. Ability to perform basic HTTP operations (GET, PUT, POST, DELETE) via some sort of programmatic tool. This document will use curl in its examples, but there are several excellent command line (wget) and browser-based tools (REST Client for Firefox, Postman and other tools for Chrome)
5. Usually searching and ordering can be done without logging in, however, some data requires a registered and active URS account (<http://urs.eosdis.nasa.gov>).

Multiple Possible Output Formats

Like many RESTful web services, ECHO supports exporting results in multiple formats. Whether you are more comfortable with a reference list, full metadata exports in various formats or simple json results, you will find what you are looking for via ECHO's REST web-service format extensions.

Document Conventions

REST call parameters are shown using the following format:

Route:	<code>https://api-test.echo.nasa.gov/<route></code>
Verb:	{GET, PUT, POST, DELETE}
Supported Extensions:	{.xml, .json, .echo10, .iso19115, ... }
Header:	Name = Value

The following styling is used for sample xml bodies and example responses:

```
<element>
<subelement>value</subelement>
<subelement>value</subelement>
</element>
```

The following styling is used for sample command line curl invocations:

- Overview
- Assumptions and Pre-requisites
- Multiple Possible Output Formats
- Document Conventions
- Currently Supported Search Metadata Formats
- Searching for Collections
 - Starting with the Simplest Case
 - Adding Parameters to the Mix
 - Finding Collections Using Spatial Constraints
 - Searching by GCMD Science Keywords
 - Retrieving a Single Collection

```
curl https://api-test.echo.nasa.gov
```

Currently Supported Search Metadata Formats

First things first, if you haven't used ECHO previously, you should probably be familiar with the output formats we support. The table below lists the current formats that can be requested via url extension.

Extension	Description	Documentation Links
.xml or none	XML including references to data piece; contains a name, id, and location of a resource.	<ul style="list-style-type: none">https://api.echo.nasa.gov/catalog-rest/catalog-docs/index.html (under complex types, "references")
.echo10	Full metadata in the ECHO10 format	<ul style="list-style-type: none">https://api.echo.nasa.gov/ingest/
.iso19115	Limited metadata in ISO 19115 format (proof of concept implementation)	<ul style="list-style-type: none">http://www.isotc211.org/2005/
.atom	Limited metadata in atom FROST format	<ul style="list-style-type: none">http://en.wikipedia.org/wiki/Atom_(standard)http://wiki.esipfed.org/index.php/Federated_Search
.json	Limited metadata in json format	<ul style="list-style-type: none">http://www.json.org/

Searching for Collections

Starting with the Simplest Case

The following few sections cover the simplest form of data discovery/metadata retrieval via ECHO's REST API: collection retrieval with no parameters or header fields set. This resource, as listed, will retrieve all publicly available collections currently indexed by the ECHO system, can be invoked via curl or wget directly and can return a variety of differently formatted responses. For this first section, we have included an exhaustive list of all valid URL extensions just to give you an idea of the breadth of options available. For successive examples, only the default reference route will be covered but don't worry, all extensions are supported for all collection searches. By default, this invocation without any parameters will return 10 references.

Remember, some collections and granules aren't available to anonymous guest users and you'll need to use an ECHO Token. See Appendix A for information on logging in and creating tokens.

Interpreting ECHO Responses

The standard response to a dataset request will return a list of references to publicly visible datasets cataloged by ECHO. Oftentimes a full metadata response is not required and can produce

- Searching for Granules
 - Retrieving a granule
 - Starting Out Simply
 - Finding Granules Using Spatial Constraints
 - Searching by Cloud Cover
 - Searching by Day/Night Flag
 - Searching by Equatorial Crossing Parameters and Orbit Number
 - Searching by Attribute
 - Retrieving a Single Granule

overly large responses so references are returned by default. The first ten results are returned sorted alphabetically by name (this can be changed by parameters).

```
curl https://api-test.echo.nasa.gov/catalog-rest/echo_catalog/datasets
```

or

```
curl https://api-test.echo.nasa.gov/catalog-rest/echo_catalog/datasets.xml
```

Excerpt from Sample Response:

```
<references type="array">
<reference>
<name>ACES CONTINUOUS DATA V1</name>
<id>C30292-GHRC</id>
<location>

https://api-test.echo.nasa.gov:443/catalog-rest/echo_catalog/datasets/C
30292-GHRC

</location>
</reference>
<reference>
<name>ACES ELECTRIC FIELD MILL V1</name>
<id>C30319-GHRC</id>
<location>

https://api-test.echo.nasa.gov:443/catalog-rest/echo_catalog/datasets/C
30319-GHRC

</location>
</reference>
...
...
```

Notice that the <reference> element contains a child element: <location>. The location field can be used to retrieve that dataset directly from ECHO via another RESTful route invocation. This document will look at individual dataset retrieval in the section entitled Retrieving a Single Collection.

Also, the <id> element is something known as the "echo_collection_id" or alternately the "catalog_item_id". This <id> will be useful during metadata retrieval and ordering. For more information about all the various ids used by ECHO and its data providers, refer to Appendix B – Name that Data: Sorting Out Short Name, Long Name and Every Identifier in Between

For any curl command you run, you can always include the '-i' flag for a more verbose response. This flag includes the HTTP-header in the response. This will tell you some useful information about your results including the HTTP return code and the number of results returned from your query. This will be discussed in more depth the section devoted to paging through large sets of results.

Expanding ECHO's Response to Include Metadata

The .echo10 extension requests that data be returned formatted using the ECHO10 format.

```
curl https://api-test.echo.nasa.gov/catalog-rest/echo_catalog/datasets.
echo10
```

Excerpt from Sample Response:

- General Searching Functionality:
Supporting Both Collections and Granules
 - Searching By Various Collection Identifiers
 - Searching by Campaign, Platform, Instrument, or Sensor
 - Paging Through Results
 - Wildcard Support
 - Finding Non-Public Data
 - Sorting Returned Data
- Appendix A - ECHO Tokens
 - Creating your token
 - Removing your token
- Appendix B – Name that Data: Sorting Out Short Name, Long Name and Every Identifier in Between
- Appendix C - Full API Documentation
- Appendix D - Ordering Catalog Items

```

<results>
<result echo_dataset_id="C30292-GHRC">
<Collection>
<ShortName>aces1cont</ShortName>
<VersionId>1</VersionId>
<InsertTime>2004-05-04T00:00:00.000Z</InsertTime>
<LastUpdate>2008-03-26T11:30:05.000Z</LastUpdate>
<LongName>ACES CONTINUOUS DATA</LongName>
<DataSetId>ACES CONTINUOUS DATA V1</DataSetId>
...
</Collection>
</result>
<result echo_dataset_id="C30319-GHRC">
<Collection>
...
</Collection>
</result>

```

- Appendix E – Granule CSV Sample Output
- Appendix F – Random Questions and Troubleshooting Guide

Date	Description	Author
11/2012	Initial Revision	Kathleen Baynes
4/2014	Port to Wiki	Kathleen Baynes

As you can see from the sample response snippet included above, each <result> element includes full xml representation of the collection in ECHO10. Documentation and .xsd files for this format can be found by referring to the table in the Currently Supported Search Metadata Formats section of this document.

ISO19115

The ISO 19115 implementation currently supported by ECHO as of this writing is subject to change. The currently supported implementation is documented and linked to by the table table in the Currently Supported Search Metadata Formats section of this document.

```
curl https://api-test.echo.nasa.gov/catalog-rest/echo_catalog/datasets.iso19115
```

Excerpt from Sample Response:

```

<results>
<result echo_dataset_id="C30292-GHRC">
<gmd:DS_Series
<gmd:seriesMetadata>
<gmi:MI_Metadata>
<gmd:fileIdentifier>
<gco:CharacterString>ACES CONTINUOUS DATA V1</gco:CharacterString>
</gmd:fileIdentifier>
<gmd:dateStamp>
<gco:DateTime>2008-03-26T11:30:05.000Z</gco:DateTime>
</gmd:dateStamp>
<gmd:identificationInfo>
<gmd:MD_DataIdentification>
<gmd:citation>
<gmd:CI_Citation>
<gmd:alternateTitle>
<gco:CharacterString>aces1cont</gco:CharacterString>
</gmd:alternateTitle>
<gmd:title>
<gco:CharacterString>ACES ...</gco:CharacterString>
</gmd:title>
<gmd:edition>
<gco:CharacterString>1</gco:CharacterString>
</gmd:edition>
<gmd:CI_Citation>
</gmd:citation>
...

```

Atom+ FROST

Atom is a well-known publishing standard. FROST stands for Federated Recursive OpenSearch Tools. Documentation for this format can be found by referring to the table in the Currently Supported Search Metadata Formats section of this document.

```
curl https://api-test.echo.nasa.gov/catalog-rest/echo_catalog/datasets.  
atom
```

Excerpt from Sample Response:

```
<feed xmlns="http://www.w3.org/2005/Atom">  
<updated>2012-08-29T14:44:12.060Z</updated>  
<id>  
https://api-test.echo.nasa.gov/catalog-rest/echo\_catalog/datasets.atom  
</id>  
<title type="text">ECHO dataset metadata</title>  
<entry xmlns:georss="http://www.georss.org/georss/1.0"  
xmlns:time="http://a9.com/-/opensearch/extensions/time/1.0/"  
xmlns:echo="http://www.echo.nasa.gov/esip"  
xmlns:gml="http://www.opengis.net/gml">  
<id>C30292-GHRC</id>  
<title>ACES CONTINUOUS DATA V1</title>  
<updated>2008-03-26T11:30:05.000Z</updated>  
<echo:datasetId>ACES CONTINUOUS DATA V1</echo:datasetId>  
<echo:shortName>aces1cont</echo:shortName>  
<echo:versionId>1</echo:versionId>  
<echo:dataCenter>GHRC</echo:dataCenter>  
<echo:archiveCenter>GHRC</echo:archiveCenter>  
<echo:processingLevelId>1B</echo:processingLevelId>  
<time:start>2002-07-10T00:00:00.000Z</time:start>  
<time:end>2002-08-30T23:59:00.000Z</time:end>  
<link href="http://aces.nsstc.nasa.gov/" hreflang="en-US" title="  
(Homepage)" rel="http://esipfed.org/ns/fedsearch/1.1./metadata#"/>  
<link  
href="ftp://ghrctest.nsstc.nasa.gov/pub/doc/aces/aces1_dataset.html"  
hreflang="en-US" title=" (Guide)"  
rel="http://esipfed.org/ns/fedsearch/1.1./metadata#"/>  
<georss:box>23.0 -85.0 26.0 -81.0</georss:box>  
<echo:difId>aces1cont</echo:difId>  
<echo:onlineAccessFlag>false</echo:onlineAccessFlag>  
<echo:browseFlag>false</echo:browseFlag>  
</entry>  
...
```

JSON

JSON (JavaScript Object Notation) is a human-readable, open standard for data exchange.

```
curl https://api-test.echo.nasa.gov/catalog-rest/echo_catalog/datasets.  
json
```

Excerpt from Sample Response:

```
{
  "feed": {
    "updated": "2012-08-29T14:45:57.646Z",
    "id": "https://api-test.echo.nasa.gov/catalog-rest/echo_catalog/datasets.json",
    "title": "ECHO dataset metadata",
    "entry": [
      {
        "id": "C30292-GHRC",
        "title": "ACES CONTINUOUS DATA V1",
        "updated": "2008-03-26T11:30:05.000Z",
        "dataset_id": "ACES CONTINUOUS DATA V1",
        "short_name": "aces1cont",
        "version_id": "1",
        "data_center": "GHRC",
        "archive_center": "GHRC",
        "processing_level_id": "1B",
        "time_start": "2002-07-10T00:00:00.000Z",
        "time_end": "2002-08-30T23:59:00.000Z",
        "links": [
          {
            "href": "http://aces.nsstc.nasa.gov/",
            "hreflang": "en-US",
            "title": "(Homepage)",
            "rel": "http://esipfed.org/ns/fedsearch/1.1./metadata#"
          },
          {
            "href": "ftp://ghrctest.nsstc.nasa.gov/pub/doc/aces/aces1_dataset.html",
            "hreflang": "en-US",
            "title": "(Guide)",
            "rel": "http://esipfed.org/ns/fedsearch/1.1./metadata#"
          }
        ],
        "boxes": [
          "23.0 -85.0 26.0 -81.0"
        ],
        "dif_ids": [
          "aces1cont"
        ],
        "online_access_flag": "false",
        "browse_flag": "false"
      },
      ...
    ]
  }
}
```

Method Summary

Route:	https://api-test.echo.nasa.gov/catalog-rest/echo_catalog/datasets
Verb:	GET
Supported Extensions:	.xml, .echo10, .iso19115, .atom, .json
Header:	None Required

Adding Parameters to the Mix

Once you are comfortable with the basic retrieval of collections using various output formats, it is time to add some parameters in place to hone in on more specific data. The examples below illustrate so more in-depth retrievals. While this is not an exhaustive example of collection discovery, it provides a solid jumping off point for creating more in-depth, specific queries.

Here is a query that will search public ECHO collections using a search string and time range (keyword and temporal parameters, respectively), and will return a specified maximum number of results (up to a maximum of 2000, specified by the "page_size" parameter). See the section of this document on Paging Through Results for more information on dealing with paging through search results.

Route:	<code>https://api-test.echo.nasa.gov/catalog-rest/echo_catalog/datasets.xml?keyword=<keyword>&page_size=<num_results>&temporal[]=<time_range></code>
Verb:	GET
Header:	None Required

```
curl -g  
"https://api-test.echo.nasa.gov/catalog-rest/echo_catalog/datasets.xml?  
keyword=Fire&page_size=7  
&temporal[ ]=2012-08-06T00:00:00Z,2012-08-10T23:59:59Z"
```

Sample Response:

```
<references type="array">  
<reference>  
<name>MODIS/Aqua Thermal Anomalies/Fire 5-Min L2 Swath 1km V005</name>  
<id>C28288-AETD8</id>  
<location>  
https://api-test.echo.nasa.gov:443/catalog-rest/echo\_catalog/datasets/C28288-AETD8  
</location>  
</reference>  
<reference>  
<name>MODIS/Aqua Thermal Anomalies/Fire 5-Min L2 Swath 1km V005</name>  
<id>C1103-LPDAAC_TS1</id>  
<location>  
https://api-test.echo.nasa.gov:443/catalog-rest/echo\_catalog/datasets/C1103-LPDAAC\_TS1  
</location>  
</reference>  
<reference>  
<name>MODIS/Aqua Thermal Anomalies/Fire 5-Min L2 Swath 1km V005</name>  
<id>C1097-LPDAAC_TS2</id>  
<location>  
https://api-test.echo.nasa.gov:443/catalog-rest/echo\_catalog/datasets/C1097-LPDAAC\_TS2  
</location>  
</reference>  
<reference>  
<name>MODIS/Aqua Thermal Anomalies/Fire 8-Day L3 Global 1km SIN Grid V005</name>  
<id>C1105-LPDAAC_TS1</id>  
<location>  
https://api-test.echo.nasa.gov:443/catalog-rest/echo\_catalog/datasets/C1105-LPDAAC\_TS1  
</location>  
</reference>  
<reference>  
<name>MODIS/Aqua Thermal Anomalies/Fire 8-Day L3 Global 1km SIN Grid V005</name>  
<id>C7683-AETD8</id>  
<location>
```

```

https://api-test.echo.nasa.gov:443/catalog-rest/echo_catalog/datasets/C
7683-AETD8

</location>
</reference>
<reference>
<name>MODIS/Aqua Thermal Anomalies/Fire 8-Day L3 Global 1km SIN Grid
V005</name>
<id>C1101-LPDAAC_TS2</id>
<location>

https://api-test.echo.nasa.gov:443/catalog-rest/echo_catalog/datasets/C
1101-LPDAAC_TS2

</location>
</reference>
<reference>
<name>MODIS/Aqua Thermal Anomalies/Fire Daily L3 Global 1km SIN Grid
V005</name>
<id>C1101-LPDAAC_TS1</id>
<location>

https://api-test.echo.nasa.gov:443/catalog-rest/echo_catalog/datasets/C
1101-LPDAAC_TS1

</location>
</reference>
</references>

```

An example of invoking this query via curl is shown above. There are a couple of things worth mentioning here. First, the -g flag is being used to prevent "globbing". From the curl documentation:

This option switches off the "URL globbing parser". When you set this option, you can specify URLs that contain the letters {}[] without having them being interpreted by curl itself. Note that these letters are not normal legal URL contents but they should be encoded according to the URI standard.

Source: <http://linux.die.net/man/1/curl>

Secondly, you can see that the double quotes surrounding the query allow special characters such as ':' and ',' without requiring URL encoding (':' becomes %3A and ',' becomes %2C). If you are programmatically invoking these routes there are probably specific functions that will encode the string for you.

Perl	http://www.perlhowto.com/encode_and_decode_url_strings
Java	http://docs.oracle.com/javase/6/docs/api/java/net/URLEncoder.html
Python	http://docs.python.org/library/urllib.html#utility-functions
Ruby	http://www.ruby-doc.org/stdlib-1.9.3/libdoc/uri/doc/URI/Escape.html

Finding Collections Using Spatial Constraints

FYI: There are some differences between the ways Reverb parameterizes spatial searches versus the backend ECHO implementation.

There are several options for discovering data that covers specific spatial constraints. ECHO

currently supports the following spatial constraint styles:

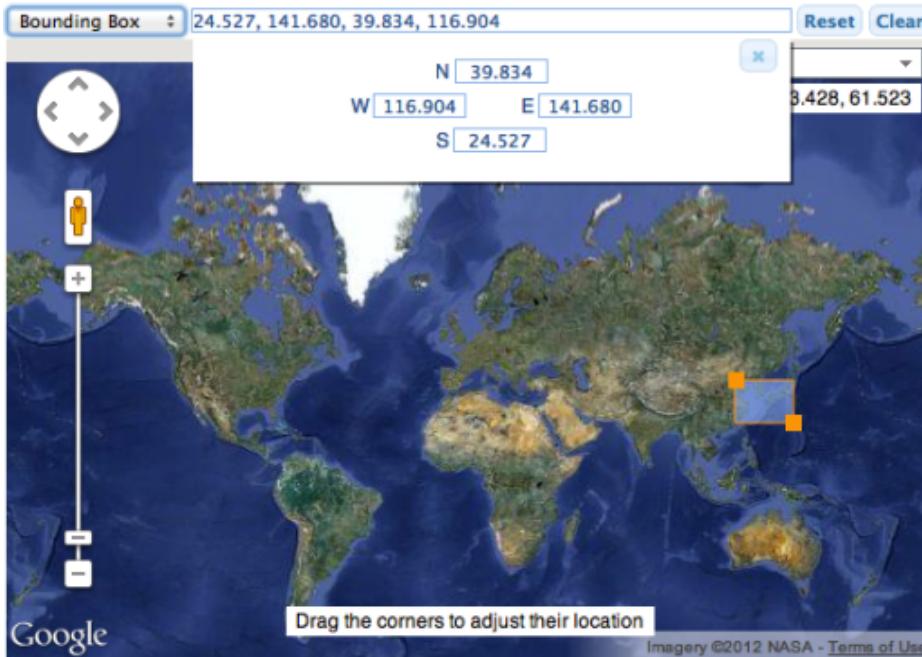
Bounding Box

Bounding box parameters must be 4 comma-separated numbers: lower left longitude, lower left latitude, upper right longitude, upper right latitude.

Sample Bounding Box Query:

```
curl  
"https://api-test.echo.nasa.gov/catalog-rest/echo_catalog/datasets.xml  
?bounding_box=116.904,24.527,141.68,39.834"
```

The screen below displays this query when performed via Reverb. Notice the coordinate order difference between Reverb and ECHO.



Source: ECHO Reverb

Point

ECHO can perform spatial queries involving a single point specified in a longitude, latitude format.
Sample Point Query:

```
curl  
"https://api-test.echo.nasa.gov/catalog-rest/echo_catalog/datasets.xml?  
point=2.295,48.858"
```

Polygon

The ECHO API documentation really covers this best:

Polygon parameters are comma-separated numbers corresponding to the first point's longitude, the first point's latitude, the second point's longitude, the second point's latitude, and so forth. Polygon points are specified in counter-clockwise order. Each polygon must have a minimum of 3 unique points, and the first point

must be the same as the last point. Therefore, every polygon consists of at least 8 comma-separated numbers.

Source: https://api.echo.nasa.gov/catalog-rest/catalog-docs/resource_datasets_datasets.html

Sample Polygon Query:

```
curl  
"https://api-test.echo.nasa.gov/catalog-rest/echo_catalog/datasets.xml?  
  
polygon=5.273,33.724,15.82,9.796,51.68,1.406,73.477,30.145,43.242,40.44  
7,5.273,33.724"
```

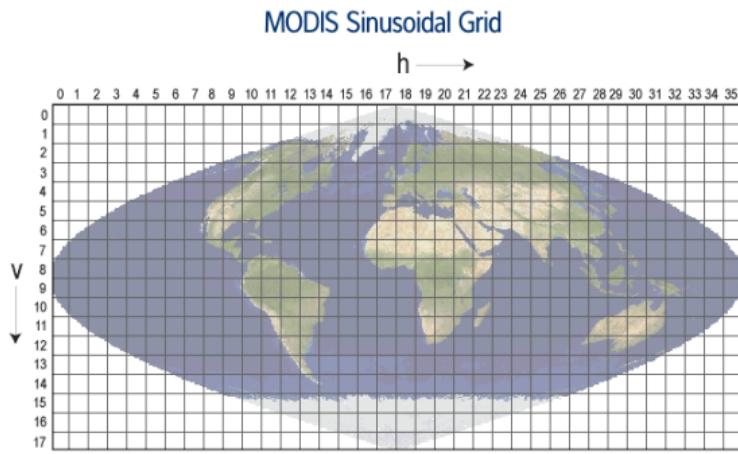
The screen below displays this query when performed via Reverb. Notice the coordinate order difference between Reverb and ECHO.



Source: ECHO Reverb

Two Dimensional Coordinate Systems Types

This method of searching will return data that has a specific type of two-dimensional coordinate data associated with its granules. For example, MODIS data is tiled and may be using differing projections. The figure below shows an example of a MODIS grid on a sinusoidal projection.



Source: http://nsidc.org/data/docs/daac/mod10_modis_snow/landgrid.html

Sample Query:

```
curl -gv
"https://api-test.echo.nasa.gov/catalog-rest/echo_catalog/datasets.xml?
two_d_coordinate_system_name[ ]=MODIS+Tile+SIN"
```

This parameter, while not codified within ECHO, does support wildcard searching.

Narrowing your search to include specific tiles is part of granule level search and discovery. See the section of this document *Finding Granules Using Spatial Constraints* for more information

Searching by GCMD Science Keywords

GCMD has a powerful list of keywords the data providers often include in their metadata to make it universally accessible to the international community.

For a list of science keywords please refer to GCMD's Science Keywords documentation:
http://gcmd.gsfc.nasa.gov/Resources/valids/archives/GCMD_Science_Keywords.pdf

ECHO supports several options when performing this search. The most common options are summarized in this table:

Parameter Option	Description
<code>science_keywords[][][topic]</code>	string representing the Topic Keyword (optional)
<code>science_keywords[][][term]</code>	string representing the Term Keyword (optional)
<code>science_keywords[][][variable_level_1]</code>	string representing the Variable Level 1 Keyword (optional)
<code>science_keywords[][][variable_level_2]</code>	string representing the Variable Level 2 Keyword (optional)
<code>science_keywords[][][variable_level_3]</code>	string representing the Variable Level 3 Keyword (optional)
<code>science_keywords[][][any]</code>	string representing any of the valid science keywords above (optional)

science_keywords[or]	boolean representing the conjunction of science keywords groups. If set to "true" science keyword searches will find datasets that match any of the science keyword groups. (default:false)
-----------------------------	---

Below is an example of a science keyword search that looks for any metadata that includes the topic "CRYOSPHERE" which returns 103 collections at the time of this writing.

Sample Query:

```
curl -g  
"https://api-test.echo.nasa.gov/catalog-rest/echo_catalog/datasets.xml?  
science_keywords[0][topic]=CRYOSPHERE"
```

The ECHO catalog-rest API documentation covers Science Keyword search usage in a little more detail and has examples showing more elaborate search combinations. Please refer to it for more information. (https://api.echo.nasa.gov/catalog-rest/catalog-docs/resource_datasets.html)

Retrieving a Single Collection

The previous sections covered discovering any ECHO collections that fulfill specified search criteria, but sometimes it may be required to retrieve a specific collection. Reflecting back to the default response returned from ECHO's search routes (using either .xml or no URL extension), each reference returned contains a URL that can be used to directly retrieve the referenced collection.

```
<reference>  
  
<name>MODIS/Aqua Thermal Anomalies/Fire 8-Day L3 Global 1km SIN Grid  
V005</name>  
  
<id>C1101-LPDAAC_TS2</id>  
  
<location>  
  
https://api-test.echo.nasa.gov:443/catalog-rest/echo\_catalog/datasets/C1101-LPDAAC\_TS2  
  
</location>  
</reference>
```

We can extract the "location" element from the xml response snippet above and use that URL directly in a curl command as follows:

```
curl https://api-test.echo.nasa.gov:443/catalog-rest/echo\_catalog/datasets/C1101-LPDAAC\_TS2
```

Response snippet:

```

<Collection>
<ShortName>MYD14A2</ShortName>
<VersionId>5</VersionId>
<InsertTime>2005-01-14T09:14:00.013Z</InsertTime>
<LastUpdate>2009-08-11T11:05:55.303Z</LastUpdate>
<LongName>MODIS/Aqua Thermal Anomalies/Fire 8-Day L3 Global 1km SIN Grid</LongName>
<DataSetId>MODIS/Aqua Thermal Anomalies/Fire 8-Day L3 Global 1km SIN Grid V005</DataSetId>
<Description>MODIS/Aqua Thermal Anomalies/Fire 8-Day L3 Global 1km SIN Grid</Description>
<Orderable>false</Orderable>
<Visible>true</Visible>
<RevisionDate>2009-06-18T00:00:00.000Z</RevisionDate>
<SuggestedUsage>Science Research</SuggestedUsage>
<ProcessingCenter>MODAPS</ProcessingCenter>
<ProcessingLevelId>3</ProcessingLevelId>
<ProcessingLevelDescription>Sensor Measurements</ProcessingLevelDescription>
<ArchiveCenter>LPDAAC</ArchiveCenter>
<VersionDescription>Collection Version 5 Processing and Reprocessing</VersionDescription>
<CitationForExternalPublication>This data set was provided by the NASA EOS Project.</CitationForExternalPublication>
<CollectionState>In Work</CollectionState>
<MaintenanceAndUpdateFrequency>As Needed</MaintenanceAndUpdateFrequency>
<SpatialKeywords />
<TemporalKeywords />
<Temporal>
<TimeType>UTC</TimeType>
<DateType>Gregorian</DateType>
<TemporalRangeType>Continuous Range</TemporalRangeType>
<PrecisionOfSeconds>1</PrecisionOfSeconds>
<EndsAtPresentFlag>true</EndsAtPresentFlag>
<RangeDateTime>
<BeginningDateTime>2002-05-04T00:00:00.000Z</BeginningDateTime>
<EndingDateTime>2017-05-04T00:00:00.000Z</EndingDateTime>
</RangeDateTime>
...

```

Notice that this response is using the ECHO10 format by default. This collection was ingested into ECHO using ECHO10. Any collection imported using a different format, will be returned, by default in that original, pristine format. ECHO supports other formats for this response, retrieved by simply appending the desired URL extension to the path component, before any optional keywords. The following table summarizes the general format of collection retrieval including supported extensions.

Route:	<a href="https://api-test.echo.nasa.gov/catalog-rest/echo_catalog/datasets/<echo collection id>">https://api-test.echo.nasa.gov/catalog-rest/echo_catalog/datasets/<echo collection id>
Verb:	GET
Supported Extensions:	.xml, .iso19115, .json, .atom
Header:	Not Required

Searching for Granules

Retrieving a granule

Narrowing your search down to specific collection can greatly increase the utility of your granule search results. However, to retrieve all granules (106 million and counting) you can use this

Method Outline:

Route:	https://api-test.echo.nasa.gov/catalog-rest/echo_catalog/granules
Verb:	GET
Header:	Not Required

Starting Out Simply

Because there are over 100 million granules currently indexed by ECHO, it probably will never make sense to do a query against all of them. This simple start assumes you have a collection in mind.

```
curl -g
"https://api-test.echo.nasa.gov:443/catalog-rest/echo_catalog/granules?
echo_collection_id[ ]=C1107-LPDAAC_TS1"
```

The value "echo_collection_id" parameter can be retrieved from the reference returned from a collection search, specifically the <id> element.

Response snippet:

```
<references type="array">
<reference>
<name>SC:MYD15A2.005:691333</name>
<id>G177867-LPDAAC_TS1</id>
<location>

https://api-test.echo.nasa.gov:443/catalog-rest/echo\_catalog/granules/G177867-LPDAAC\_TS1

</location>
</reference>
<reference>
<name>SC:MYD15A2.005:684299</name>
<id>G186030-LPDAAC_TS1</id>
<location>

https://api-test.echo.nasa.gov:443/catalog-rest/echo\_catalog/granules/G186030-LPDAAC\_TS1

</location>
</reference>
<reference>
<name>SC:MYD15A2.005:690361</name>
<id>G188278-LPDAAC_TS1</id>
<location>

https://api-test.echo.nasa.gov:443/catalog-rest/echo\_catalog/granules/G188278-LPDAAC\_TS1

</location>
</reference>
...
```

Similar to default results returned by the collections query, this shows references to granule metadata using the <location> element included in each <reference>. You can use these URLs to directly retrieve a granule in the ECHO10 format.

Finding Granules Using Spatial Constraints

Narrowing your granule search to a geographical area can be done in a similar manner to

collections. There are options for searching by bounding box, point, and polygon. If you are interested in doing these kinds of searches, please refer to the Collections section of this document for examples and simply replace the "datasets" with granules in any of the curl commands. However, there is one additional spatial style search that is not supported at the collection level and can only be utilized when performing granule queries: two-dimensional coordinate system parameters. Note, the collection level search supports searching for **names** for two-dimensional coordinate systems (e.g. wrs-1), but does not support spatial boundary values.

More specific information on this search is best served by the API documentation itself:

At least one bound from each coordinate set must be supplied. For example, a search for granules with a coordinate scheme wrs-1 with an x coordinate in the range of 0-5 and a y coordinate in the range of 0-10 is represented by:

```
two_d_coordinate_system[name]=wrs-1
&
two_d_coordinate_system[x_min]=0
&
two_d_coordinate_system[x_max]=5
&
two_d_coordinate_system[y_min]=0
&
two_d_coordinate_system[y_max]=10.
```

A search for granules with a coordinate scheme wrs-1 with an x coordinate greater than or equal to 0 and a y coordinate in the range of 0-10 is represented by:

```
two_d_coordinate_system[name]=wrs-1
&
two_d_coordinate_system[x_min]=0
&
two_d_coordinate_system[y_min]=0
&
two_d_coordinate_system[y_max]=10
```

_Source: https://api.echo.nasa.gov/catalog-rest/catalog-docs/resource_granules_granules.html_

Sample Two-Dimensional Coordinate System Query:

```
curl -ig
"https://api-test.echo.nasa.gov/catalog-rest/echo_catalog/granules.xml?
&two_d_coordinate_system[name]=MODIS+Tile+SIN
&two_d_coordinate_system[x_max]=14&two_d_coordinate_system[x_min]=10
&two_d_coordinate_system[y_max]=12&two_d_coordinate_system[y_min]=8"
```

Searching by Cloud Cover

Many granules have cloud cover percentages associated. The ECHO REST API allows queries that specify both minimum and maximum values for cloud cover. If a granule has no value for cloud cover, that granule will not be returned as a part of these results. Valid values are 0 to 100 (float).

Sample Search for 'DAY' Granules:

```
curl -ig
"https://api-test.echo.nasa.gov/catalog-rest/echo_catalog/granules?clou
d_cover[min]=0&cloud_cover[max]=1"
```

Searching by Day/Night Flag

A day/night flag can be used search for granules in ECHO. Valid values are "DAY", "NIGHT", and "BOTH"

Sample Search for 'DAY' Granules:

```
curl -ig  
"https://api-test.echo.nasa.gov/catalog-rest/echo_catalog/granules?day_  
night_flag=DAY"
```

Searching by Equatorial Crossing Parameters and Orbit Number

While these parameters are largely self-explanatory, there might be some confusion as to the implementation of the start and end dates.

To be clear, the start date should be seen as a "no earlier than" while end data should be seen as "no later than". There are validation checks to ensure that the start date specified is earlier than the end date.

Sample Search using several parameters:

```
curl -ig  
"https://api-test.echo.nasa.gov/catalog-rest/echo_catalog/granules?  
equator_crossing_longitude[min]=21  
&equator_crossing_longitude[max]=31  
&equator_crossing_start_date=2012-04-01T00:00:00Z  
&equator_crossing_end_date=2012-04-03T23:59:59Z  
&orbit_number[minValue]=65370"
```

Parameter[field]	Example URL
<code>equator_crossing_longitude[min]</code>	https://api-test.echo.nasa.gov/catalog-rest/echo_catalog/granules?equator_crossing_longitude[min]=21
<code>equator_crossing_longitude[max]</code>	https://api-test.echo.nasa.gov/catalog-rest/echo_catalog/granules?equator_crossing_longitude[max]=22
<code>equator_crossing_start_date</code>	https://api-test.echo.nasa.gov/catalog-rest/echo_catalog/granules?equator_crossing_start_date=2011-01-01T00:00:00Z
<code>equator_crossing_end_date</code>	https://api-test.echo.nasa.gov/catalog-rest/echo_catalog/granules?equator_crossing_end_date=2012-05-01T23:59:59Z
<code>orbit_number[value]</code>	https://api-test.echo.nasa.gov/catalog-rest/echo_catalog/granules?orbit_number[value]=65000
<code>orbit_number[minValue]</code>	https://api-test.echo.nasa.gov/catalog-rest/echo_catalog/granules?orbit_number[minValue]=45000
<code>orbit_number[maxValue]</code>	https://api-test.echo.nasa.gov/catalog-rest/echo_catalog/granules?orbit_number[maxValue]=55000

Searching by Attribute

Additional attributes are parameters, also known as Provider-Specific Attributes (PSAs), which further describe the data represented in each granule within a collection. These values are

important search criteria for the granules. Example attributes include values for Quality Assurance Data, MODIS Tile grid coordinates, and elevation information. All additional attribute definitions are included in the collection metadata. A collection may also specify a value, to be understood as the value for all granules, however, an individual granule may override a single value provided by the parent collection. Granules reference defined additional attributes and supply a value that is associated to that granule. Granules may not define a new additional attribute that is not defined by the collection.

The attribute parameter is probably one of the more difficult parameters to use unless you already have values in mind for query. This parameter must be used with [name], [type], [minValue], [maxValue] or [value] fields.

Parameter[field]	Description
attribute[] [name]	Name of the attribute to search
attribute[] [type]	Attribute type. (see table below)
attribute[] [minValue]	Valid for types that support range searches (see table below)
attribute[] [maxValue]	Valid for types that support range searches (see table below)
attribute[] [value]	Searches for the exact value of the attribute

Summarizing Additional Attribute Types:

Additional Attribute Type	XML Type	Sample Value
string	string	YES
float	float	0.988
int	Int	42
date	date	2011-03-28
time	time	12:59:59
dateTime	dateTime	2010-01-01T00:00:00
date_string	string	Same as date
time_string	string	Same as time
dateTime_string	string	Same as dateTime

Sample Searches:

```
curl -ig
"https://api-test.echo.nasa.gov/catalog-rest/echo_catalog/granules?
attribute[] [name]=GRANULENUMBER
&attribute[] [type]=int
&attribute[] [value]=116"
```

```
curl -ig
"https://api-test.echo.nasa.gov/catalog-rest/echo_catalog/granules?
attribute[] [name]=QA FRACTION NOT PRODUCED CLOUD&attribute[] [type]=float
&attribute[] [minValue]=0.949
&attribute[] [maxValue]=0.9495"
```

```
curl -ig  
"https://api-test.echo.nasa.gov/catalog-rest/echo_catalog/granules?  
attribute[ ][name]=CLOUD_CONTAMINATED_LST_SCREENED  
&attribute[ ][type]=string&attribute[ ][value]=NO"
```

Retrieving a Single Granule

The previous sections covered discovering any ECHO granules that fulfill specified search criteria, but sometimes it may be required to retrieve a specific granule's metadata. Reflecting back to the default response returned from ECHO's resources (using either .xml or no URL extension), each reference returned contains a URL that can be used to directly retrieve the referenced granule's metadata. The online access URL (if present in the metadata) allows you to retrieve the actual granule.

```
<reference>  
<name>SC:MYD15A2.005:684299</name>  
<id>G186030-LPDAAC_TS1</id>  
<location>  
https://api-test.echo.nasa.gov:443/catalog-rest/echo\_catalog/granules/G186030-LPDAAC\_TS1  
</location>  
</reference>
```

Just as in the collection case, we can extract the "location" element from the xml response snippet above and use that URL directly in a curl command as follows:

```
curl  
https://api-test.echo.nasa.gov:443/catalog-rest/echo\_catalog/granules/G186030-LPDAAC\_TS1
```

Response snippet:

```
<Granule>  
<GranuleUR>SC:MYD15A2.005:684299</GranuleUR>  
<InsertTime>2007-09-07T11:35:37.693Z</InsertTime>  
<LastUpdate>2012-08-21T16:28:31.193Z</LastUpdate>  
<Collection>  
<DataSetId>  
MODIS/Aqua Leaf Area Index/FPAR 8-Day L4 Global 1km SIN Grid V005  
</DataSetId>  
</Collection>  
...  
<FileSize>80415</FileSize>  
<MimeType>application/x-hdfeos</MimeType>  
<ProviderBrowseUrl>  
</AssociatedBrowseImageUrls>  
</Granule>
```

Notice that this response is using the ECHO10 format by default. This is because the granule was original ingested using that format. ECHO will return granules using the same format in which they were ingested.

ECHO supports other formats for this response, retrieved by simply appending the desired URL extension to the path component before any optional parameters. The following table summarizes the general format of granule retrieval including supported extensions. Note that granule level searching includes support for .csv (comma separated value) formats. Please see Appendix E – Granule CSV Sample Output for detail about granule .csv export.

Route:	<a href="https://api-test.echo.nasa.gov/catalog-rest/echo_catalog/granules/<granule id>">https://api-test.echo.nasa.gov/catalog-rest/echo_catalog/granules/<granule id>
Verb:	GET
Supported Extensions:	.xml, .iso19115, .json, .atom, .csv
Header:	Not Required

General Searching Functionality: Supporting Both Collections and Granules

Searching By Various Collection Identifiers

If you know which collection interests you, it is possible to search by short name, dataset id, or ECHO Collection Id at both the collection and granule level (for explanation of various ids used by ECHO, please refer to Appendix B – Name that Data: Sorting Out Short Name, Long Name and Every Identifier in Between)

Name	Parameter	Example curl
Dataset ID	dataset_id[]	curl -g "https://api-test.echo.nasa.gov/catalog-rest/echo_catalog/datasets?dataset_id[]=%2FTerra%2BAqua%20Land%20Cover%20Type%20Yearly%20L3%20Global%20500m%20SIN%20Grid%20V051"
ECHO Collection ID	echo_collection_id[]	curl -ig "https://api-test.echo.nasa.gov/catalog-rest/echo_catalog/granules?echo_collection_id[]=%20341849-LPDAAC_TS1"
Short Name (not guaranteed to be unique)	short_name[] (collection) or short_name (granule)	curl -ig "https://api-test.echo.nasa.gov/catalog-rest/echo_catalog/datasets?short_name[]=%20MCD12Q1" or curl -ig "https://api-test.echo.nasa.gov/catalog-rest/echo_catalog/granules?short_name=%20MCD12Q1"

Searching by Campaign, Platform, Instrument, or Sensor

Most ECHO collections and granules contain information on the systems used to collect the data. This data can include information on the campaign, platform, instrument, or sensor used for the data collection. All of these are searchable fields within ECHO at **both** the collection and granule levels.

The table below summarizes the usage of these parameters:

Name	Parameter	Example URL
Campaign	campaign[]	https://api-test.echo.nasa.gov/catalog-rest/echo_catalog/datasets?campaign[]=%20Aqua

Platform	platform[]	https://api-test.echo.nasa.gov/catalog-rest/echo_catalog/granules?platform[]=CALIPSO
Instrument	instrument[]	https://api-test.echo.nasa.gov/catalog-rest/echo_catalog/datasets?instrument[]=LIDAR
Sensor	sensor[]	https://api-test.echo.nasa.gov/catalog-rest/echo_catalog/granule?sensor[]=2.3UM%20RADIMETER

To search across multiple values, just include the parameter multiple times. For example, use "campaign[]=Aqua&campaign[]=Terra" to get datasets from both the Aqua and Terra campaigns.

Case Sensitivity Options

By default all campaign, platform, instrument, and sensor searches are case-sensitive. However, it is possible to specify that ECHO should ignore case mismatch and return the same results whether "AQUA" or "Aqua" is specified in the search parameter. In order to accomplish this, an additional parameter may be specified as shown. Each parameter can individually specify this option.

Sample Query:

```
curl -g
"https://api-test.echo.nasa.gov/catalog-rest/echo_catalog/datasets.echo
10?

campaign[]=calipso&options[campaign][ignore_case]=true"
```

Paging Through Results

If we run a query that returns a large number of results, we can page through these requests using parameters on our search. When a query is returned, there are a number of values returned as headers, one of which is "Echo-Hits". This returns either an exact or estimated number of returned results (specified by the header value "Echo-Hits-Estimated")

Header Name	Description	Sample Value
Echo-Hits	The total number of possible hits that match this search.	64
Echo-Hits-Estimated	Flag that describes whether or not the hits count has been estimated or changed from the original hits total.	true
Echo-Cursor-At-End	Flag describing whether or not the references returned represent the last page of possible results.	false

Once you know the nature of the results you can return, you can begin paging through them with the following parameters:

Parameter Name	Description
page_size	The number of datasets that should be returned from your search. Defaults to 10. The maximum is 2000.

page_num	The current page of results to retrieve. This is a number starting at 1. Defaults to 1. There is no maximum. Searching past the end of the results of your query returns no results.
-----------------	--

Basic algorithm for retrieving a large number of paged results:

Perform basic search. This is searching for all granules in the collection with ECHO internal collection id "C1000-LPDAAC_TS2", a test collection in partner test that is associated with "AST_L1A", returning 100 results per page and starting at page 1. Notice that the response reports **3234** granule results.

```
curl -gi
"https://api-test.echo.nasa.gov/catalog-rest/echo_catalog/granules.xml?
echo_collection_id[] = C1000-LPDAAC_TS2
&page_size=100
&page_num=1"
```

Sample Response Snippet with Header:

```
< HTTP/1.1 200 OK
< Server: Apache-Coyote/1.1
< Echo-Hits: 3234
< Echo-Hits-Estimated: false
< Echo-Cursor-At-End: false
< X-UA-Compatible: IE=Edge,chrome=1
< ETag: "927ec958c2741dc47c75d529fda22f2a"
< Cache-Control: max-age=0, private, must-revalidate
< X-Request-Id: 98e9117f4534e3d98ee4e9d7e280a276
< X-Runtime: 1.562000
< Date: Tue, 02 Oct 2012 14:34:33 GMT
< X-Rack-Cache: miss
< Content-Type: application/xml; charset=utf-8
< Transfer-Encoding: chunked
<
<?xml version="1.0" encoding="UTF-8"?>
<references type="array">
<reference>
<name>SC:AST_L1A.003:280928</name>
<id>G55495-LPDAAC_TS2</id>
<location>
https://api-test.echo.nasa.gov:443/catalog-rest/echo\_catalog/granules/G55495-LPDAAC\_TS2
</location>
</reference>
<reference>
<name>SC:AST_L1A.003:280929</name>
<id>G55496-LPDAAC_TS2</id>
<location>
https://api-test.echo.nasa.gov:443/catalog-rest/echo\_catalog/granules/G55496-LPDAAC\_TS2
</location>
</reference>
...
...
```

While the response header "Echo-Cursor-At-End" is not "true", continue the same query, incrementing the page number to retrieve successive results. For the query above, this should require 33 successive calls (the ceiling of 3234/100).

```

curl -gi
"https://api-test.echo.nasa.gov/catalog-rest/echo_catalog/granules.xml?
echo_collection_id[ ]=C1000-LPDAAC_TS2
&page_size=100
&page_num=2"

```

A few words of advice

If you want to retrieve all results use the maximum page size necessary to minimize the number of searches required. If you are presenting results to a user who will likely not visit all possible matches, use the minimum page size reasonable for your application to minimize search and result preparation time.

Be aware that large page sizes with full metadata responses can result in a lot of metadata. Take care to implement efficient parsing mechanisms when doing client development.

Wildcard Support

For many of the ECHO parameters searches, wildcard searches are permitted. Two types of wildcard are supported in these parameter searches:

Character	Description
%	Substitute zero or more characters
_ (underscore symbol)	Substitute exactly one character

To search for exact match of the character % or _, escape the wildcard character with '\'. The following common parameters support wildcard searches:

Parameter Name	Collection or Granule Level
two_d_coordinate_system_name[]	Collection
keyword	Collection
processing_level[]	Collection
science_keywords[]	Collection
provider	Both Collection and Granule
dataset_id[]	Both Collection and Granule
campaign[]	Both Collection and Granule
platform[]	Both Collection and Granule
instrument[]	Both Collection and Granule
sensor[]	Both Collection and Granule
short_name	Granule
version	Granule
granule_ur[]	Granule
producer_granule_ur[]	Granule
day_night_flag	Granule

Sample Query:

```
curl -g "https://api-test.echo.nasa.gov/catalog-rest/echo_catalog/datasets.echo10?  
campaign[]="calipso&options[campaign][ignore_case]=true"
```

Finding Non-Public Data

Because not all data in ECHO is marked as publicly available, there may be some cases in which an authentication token needs to be provided to access the desired data. Providing the header parameter "Echo-Token" and a corresponding authentication token will ensure that all data available to the authenticated users is returned from a search.

To retrieve an ECHO token that may be used in the "Echo-Token" header parameter, please refer to Appendix A - ECHO Tokens of this document.

Sorting Returned Data

There is a "sort_key" parameter defined for both Collection and Granule level searches that allows a user to sort results in ascending or descending order based on various fields within the data. The default sort order is ascending, but prepending a '-' to the field name will change sort order to descending.

Sample Query sorting collections by start_date (descending):

```
curl -g  
"https://api-test.echo.nasa.gov/catalog-rest/echo_catalog/datasets.echo10?sort_key[]=-start_date"
```

Sample Query sorting granules by cloud_cover (ascending):

```
curl -g  
"https://api-test.echo.nasa.gov/catalog-rest/echo_catalog/granules.echo10?echo_collection_id[]="C1000-LPDAAC_TS2&sort_key[]=cloud_cover"
```

Sort Field Name	Collection or Granule Level
dataset_id	Both Collection and Granule
start_date	Both Collection and Granule
end_date	Both Collection and Granule
platform	Both Collection and Granule
instrument	Both Collection and Granule
sensor	Both Collection and Granule
campaign	Granule
granule_ur	Granule

<code>producer_granule_id</code>	Granule
<code>cloud_cover</code>	Granule
<code>day_night_flag</code>	Granule
<code>short_name</code>	Granule
<code>version</code>	Granule

Appendix A - ECHO Tokens

Creating your token

See the following page: [Creating a Token](#)

Removing your token

While ECHO encourages you to reuse tokens when feasible, if you are finished with your token, it is nice to clean up after yourself. If you are performing programmatic discovery and retrieval and want to generate a new token for each batch of queries you are doing, we recommend this best practice of deleting your token. This will invalidate your token.

Be aware that if a client will be interacting on behalf of end users that might have differing user permissions, that client should use different tokens for each user.

Method Outline:

Route:	<a href="https://api-test.echo.nasa.gov/echo-rest/tokens/<token to delete>">https://api-test.echo.nasa.gov/echo-rest/tokens /<token to delete>
Verb:	DELETE
Header:	Content-Type = application/xml
Header:	Echo-Token = <token created above>

Appendix B – Name that Data: Sorting Out Short Name, Long Name and Every Identifier in Between

Sometimes the number of identifiers ECHO employs can seem a bit overwhelming. This section aims to define and clarify several terms used in identifying an individual collection or granule within ECHO.

Identifier Name	Parameter	Level	Description
-----------------	-----------	-------	-------------

Catalog Item ID	<i>varies</i>	Both	This is a unique identifier generated by ECHO to uniquely identify a metadata record. This id takes the form: XNNNNN-PROV_NAME, where X is either C or G (for collection or granule), NNNNN is a number (not necessarily 5 digits), and PROV_NAME is the ECHO representation of the provider's name. (Example: C184964546-LARC)
Data Set Id	dataset_id[]	Collection	This attribute specifies a unique name for the collection. This value is often built by concatenating the Collection Long Name with a version number (Example: ASTER Expedited L1A Reconstructed Unprocessed Instrument Data V003)
ECHO Collection Id	echo_collection_id[]	Collection	Synonymous with catalog item id for collections, prefixed with C. (Example: C184964546-LARC)
Collection Short Name	short_name	Collection	This attribute identifies the short name associated with the collection. This is the official reference name used in identifying the contents of the data collection. This is not necessarily unique. (Example: AST_L1A)
Collection Long Name		Collection	This attribute will identify the long name associated with the collection. This is the reference name used in describing the scientific contents of the data collection. (Example: GHRSST-PP L2P Sea Surface Temperature SSTskin observations from the ENVISAT Advanced Along Track Scanning Radiometer (AATSR))
Granule UR	granule_ur[]	Granule	The Universal Reference (UR) ID of the granule referred to by the data provider. This ID is unique per data provider. (Example: SC:AE_SI25.002:32676925)
ECHO Granule Id	echo_granule_id[]	Granule	Synonymous with catalog item id for granules, prefixed with G. (Example: G188664515-LARC)

Producer Granule Id	producer_granule_id[]	Granule	The producer of the granule, often a science team, generates this id. It is not guaranteed to be unique. If available this is the id shown in Reverb granule results list. (Example: AMSR_E_L3_Sealce25km_V12_20020619.hdf)
---------------------	------------------------------	---------	---

Appendix C - Full API Documentation

For a dictionary of methods available via our REST interfaces, please refer to the following links.

<https://api.echo.nasa.gov/catalog-rest/catalog-docs/index.html>
<https://api.echo.nasa.gov/catalog-rest/ingest-docs/index.html>
<https://api.echo.nasa.gov/echo-rest/docs/index.html>

Appendix D - Ordering Catalog Items

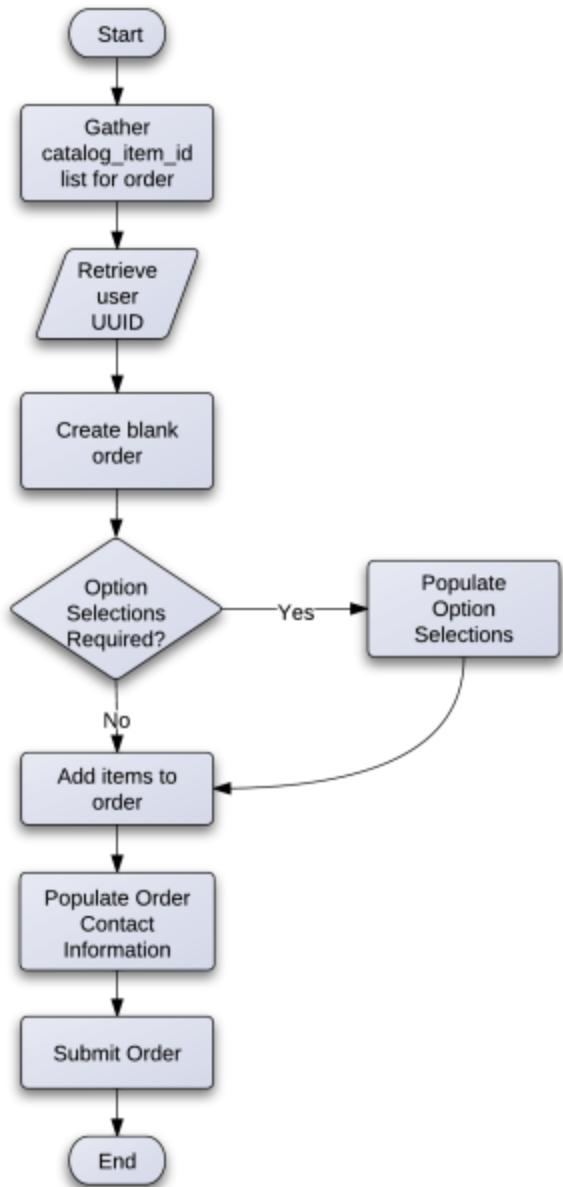
This document is concerned with searching ECHO's catalog holdings, however you will find a high level outline of the ordering process below.

There are various steps related to generating and placing orders via providers. Not all providers support an ordering interface.

Each of these providers follows its own maintenance schedule and may fulfill orders at different rates. ECHO does not fulfill data orders; it simply facilitates the creation and submission of these orders.

1. Gather catalog_item_id list including items to be ordered
2. Retrieve a token
3. Retrieve user id (UUID)
4. Generate a blank order, retrieving order id (UUID)
5. Retrieve required option selections (if any)
6. Add items to order, including option selections as escaped XML
7. Set user contact information for ordering
8. Submit order for validation and processing

Figure 1 Generic Order Workflow



Appendix E – Granule CSV Sample Output

Granule level data export supports the comma separated value (.csv) extension for requests. The following request will show the first 10 granules from the GHRC collection with ECHO collection ID "C514862-GHRC".

```
curl -g
"https://api.echo.nasa.gov/catalog-rest/echo_catalog/granules.csv?echo_
collection_id[]="C514862-GHRC"
```

The response is included on the following page.
 NOTE: Blank fields indicate that no data was provided for that field in the metadata.

Granule UR	Producer Granule ID	Start Time	End Time	Online Access URLs	Browse URLs	Cloud Cover	Day/Night	Size
f08_ssni_1 98707v7.nc		1987-07-01 T00:00:00. 000Z	1987-07-31 T23:59:59. 000Z	ftp://ghrc.n sstc.nasa.g ov/pub/ssm i/f08/monthl y/data/198 7/f08_ssni _198707v7 .nc			UNSPECIF IED	1.2607250 21
f08_ssni_1 98708v7.nc		1987-08-01 T00:00:00. 000Z	1987-08-31 T23:59:59. 000Z	ftp://ghrc.n sstc.nasa.g ov/pub/ssm i/f08/monthl y/data/198 7/f08_ssni _198708v7 .nc			UNSPECIF IED	1.2105846 41
f08_ssni_1 98709v7.nc		1987-09-01 T00:00:00. 000Z	1987-09-30 T23:59:59. 000Z	ftp://ghrc.n sstc.nasa.g ov/pub/ssm i/f08/monthl y/data/198 7/f08_ssni _198709v7 .nc			UNSPECIF IED	1.1981830 6
f08_ssni_1 98710v7.nc		1987-10-01 T00:00:00. 000Z	1987-10-31 T23:59:59. 000Z	ftp://ghrc.n sstc.nasa.g ov/pub/ssm i/f08/monthl y/data/198 7/f08_ssni _198710v7 .nc			UNSPECIF IED	1.2224445 34
f08_ssni_1 98711v7.nc		1987-11-01 T00:00:00. 000Z	1987-11-30 T23:59:59. 000Z	ftp://ghrc.n sstc.nasa.g ov/pub/ssm i/f08/monthl y/data/198 7/f08_ssni _198711v7 .nc			UNSPECIF IED	1.2022256 85
f08_ssni_1 98712v7.nc		1987-12-01 T00:00:00. 000Z	1987-12-31 T23:59:59. 000Z	ftp://ghrc.n sstc.nasa.g ov/pub/ssm i/f08/monthl y/data/198 7/f08_ssni _198712v7 .nc			UNSPECIF IED	1.3629417 42
f08_ssni_1 98801v7.nc		1988-01-01 T00:00:00. 000Z	1988-01-31 T23:59:59. 000Z	ftp://ghrc.n sstc.nasa.g ov/pub/ssm i/f08/monthl y/data/198 8/f08_ssni _198801v7 .nc			UNSPECIF IED	1.2993640 9
f08_ssni_1 98802v7.nc		1988-02-01 T00:00:00. 000Z	1988-02-29 T23:59:59. 000Z	ftp://ghrc.n sstc.nasa.g ov/pub/ssm i/f08/monthl y/data/198 8/f08_ssni _198802v7 .nc			UNSPECIF IED	1.2283744 81
f08_ssni_1 98803v7.nc		1988-03-01 T00:00:00. 000Z	1988-03-31 T23:59:59. 000Z	ftp://ghrc.n sstc.nasa.g ov/pub/ssm i/f08/monthl y/data/198 8/f08_ssni _198803v7 .nc			UNSPECIF IED	1.2295293 81

f08_ssni_1 98804v7.nc		1988-04-01 T00:00:00. 000Z	1988-04-30 T23:59:59. 000Z	ftp://ghrc.n sstc.nasa.g ov/pub/ssm i/f08/monthl y/data/198 8/f08_ssni _198804v7 .nc			UNSPECIF IED	1.2141742 71
--------------------------	--	----------------------------------	----------------------------------	---	--	--	-----------------	-----------------

Appendix F – Random Questions and Troubleshooting Guide

Question

There is a "search" resource at both the dataset and granule level. What is the deal with this?

Answer

This is a legacy-style resource that can be used if you are familiar with the ECHO SOAP-style API and AQL. The "search" supports the POST verb only and takes a body of AQL (see the AQL dtd here: <https://api.echo.nasa.gov/echo/dtd/IIMSAQLQueryLanguage.dtd>).

Questions To Be Added as Needed